VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV RADIOELEKTRONIKY
DEPARTMENT OF RADIO ELECTRONICS

EMBEDDED ZPRACOVÁNÍ VIDEA PRO DOHLEDOVÝ SYSTÉM
EMBEDDED VIDEO PROCESSING FOR SURVEILLANCE SYSTEMS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Lukáš Gerych**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Tomáš Frýza, Ph.D.**
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Elektronika a sdělovací technika**
Ústav radioelektroniky

Student: Lukáš Gerych ***ID:*** 174298
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Embedded zpracování videa pro dohledový systém

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte dosavadní projekty/knihovny pro zpracování video signálů a počítačové vidění (např. OpenCV). Pro vhodnou platformu rodiny ARM navrhněte řetězec pro snímání statické scény a vyhodnocování obrazu s funkcemi pro dohledový systém, např. detekce pohybu, identifikace objektů, apod. Sestavte snímací systém a navrhněte způsob uživatelského ovládání systému, vč. logování událostí.

Oživte celý řetězec zpracování reálných video signálů a proveďte detailní testování všech funkcí.

DOPORUČENÁ LITERATURA:

[1] OpenCV: Open Source Computer Vision [online]. 2016 [cit. 2016-05-24]. Dostupné z: http://opencv.org/.

[2] Raspberry Pi [online]. Raspberry Pi Foundation, 2016 [cit. 2016-05-24]. Dostupné z: https://www.raspberrypi.org/.

Termín zadání: 6.2.2017 ***Termín odevzdání:*** 30.5.2017

Vedoucí práce: doc. Ing. Tomáš Frýza, Ph.D.
Konzultant:

prof. Ing. Tomáš Kratochvíl, Ph.D.
předseda oborové rady

ABSTRAKT

Obsahem práce je návrh a realizace vestavěného systému pro snímání statické scény a vyhodnocování obrazu pro dohledový systém. Je prováděna na miniaturním počítači Raspberry Pi s procesorem ARM. Jsou zde uvedeny metody různých typů detekcí v obraze a jejich implementace s použitím knihoven OpenCV. Dále se zabývá způsobem přístupu k systému pro jeho ovládání a prohlížení záznamů detekcí.

KLÍČOVÁ SLOVA

ARM, Raspberry Pi, jasová složka, OpenCV, počítačové vidění, zpracování obrazu, detekce pohybu, detekce obličeje

ABSTRACT

The thesis includes the design and implementation of embedded system to capture static scenes and image evaluation for the surveillance system. It is performed on a tiny computer Raspberry Pi with ARM processor. Methods are provided for various types of detection in the image and their implementation using OpenCV libraries. It also deals with ways to access the system for the control and viewing records of detections.

KEYWORDS

ARM, Raspberry Pi, luminance component, OpenCV, computer vision, image processing, motion detection, face detection

GERYCH, Lukáš *Embedded zpracování videa pro dohledový systém* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky, 2017. 50 s. Bakalářská práce. Vedoucí práce: doc. Ing. Tomáš Frýza, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svoji bakalářskou práci na téma Embedded zpracování videa pro dohledový systém jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ....

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu bakalářské práce panu doc. Ing. Tomáši Frýzovi, Ph.D. za odborné vedení, konzultace, trpělivost, cenné rady a návrhy pro zpracování mé bakalářské práce.

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Tato práce spadá do oblasti zpracování digitálního obrazu a počítačového vidění. Z velké části se věnuje programováním v jazycích C i v C++.

Cílem práce je návrh a realizace zařízení pro snímání statické scény, které by mohlo být použito jako bezpečnostní systém založený na detekci objektů z obrazu kamery.

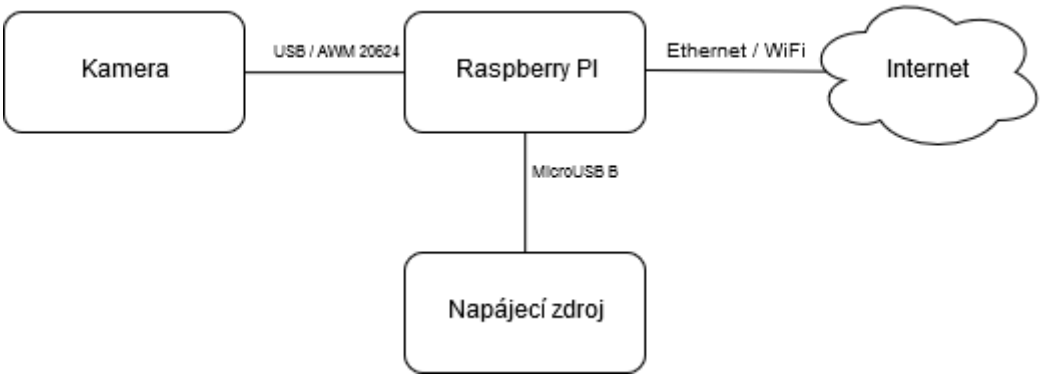
K tomu bylo potřeba prostudovat metody a algoritmy projektů zabývajících se na toto téma. Hlavně jde tedy o funkce z knihoven OpenCV, ve kterých je už několik takovýchto algoritmů předpřipraveno.

Bakalářská práce je rozdělena na 6 částí. První tři jsou představeny jako teoretická část zabývají se vlastnostmi použitého zařízení pro realizaci práce, základními informacemi o digitálním obrazu a informacemi o knihovně OpenCV, na které je práce založena. Další tři kapitoly popisují postup práce a to postup zpracování obrazu s využitím funkcí OpenCV, realizaci přístupu k systému v podobě webové stránky a testování systému z různých hledisek, podmínek a situací.

Výsledkem práce je, že systém dokáže dekovat pohyb, obličej, horní část těla nebo ruku. Po jeho spuštění se dá ovládat pomocí webové stránky běžící na webovém serveru zařízení. Taktéž zaznamenává snímky, které je možné prohlížet.

1 ZAŘÍZENÍ

Jako zařízení vestavěného systému pro zpracování obrazu je v této práci využit miniaturní počítač Raspberry Pi využívající procesor ARM. K němu bude připojena kamera pro snímání statické scény. Zařízení bude také připojeno k internetu přes WiFi nebo Ethernet pro vzdálený přístup. Celý systém znázorňuje blokové schéma na obrázku 1.1.



Obrázek 1.1 Blokové schéma systému

1.1 Procesory ARM

ARM je označení architektury procesoru z většiny 32bitového a jednotlivé architektury výrobce se označují jako ARMv3, ARMv4 apod. Jejich výhoda je v nízké spotřebě elektrické energie, v úspoře místa a levnější výrobě [1]. Díky těmto výhodám se výrazně uplatňují v mobilních zařízeních a ve vestavěných systémech.

1.1.1 Historie

Vývoj ARM procesorů začal v britské firmě ARM Holdings v 80. letech 20. století. Firma časem ustoupila od výroby procesorů a místo toho se soustředila pouze na jejich vývoj. Schéma architektur procesorů ARM je vlastnictvím firmy a od výrobců hardwaru vybírá licence za jejich použití [2].

1.1.2 Architektura

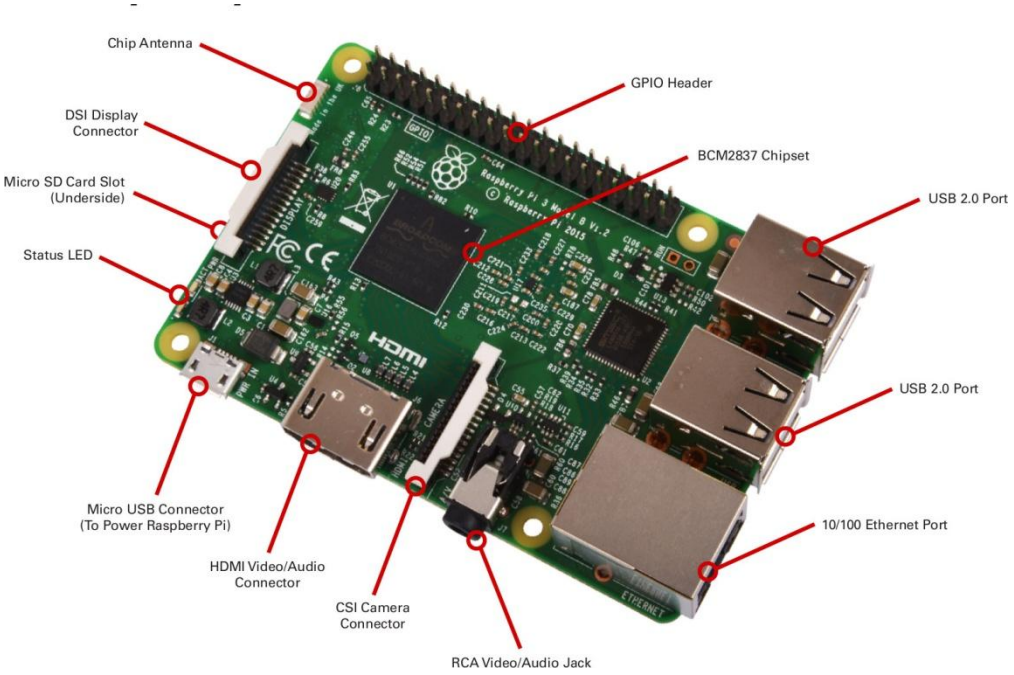
Procesor ARM obsahuje instrukce s jednotnou šířkou 32 bitů. V jednom taktu se vykonávají pouze instrukce pracující s aritmeticko-logickou jednotkou (ALU), s registry nebo s přímými operandy [2].

1.2 Raspberry Pi

Raspberry Pi je jednodeskový miniaturní počítač s deskou plošných spojů o velikosti zhruba kreditní karty. Jedná se samostatný funkční celek, tudíž neobsahuje displej a

ovládací prvky jako u mobilních telefonů, tabletů či notebooků. Jsou na něm rozmístěny jednotlivé vstupní a výstupní konektory pro připojení, tak jako u klasického stolního počítače.

Standartně obsahuje konektory USB2.0, HDMI pro připojení monitoru, microUSB pro napájení, Ethernet s RJ-45 a slot pro paměťovou kartu microSD. Novější verze Raspberry Pi 3 obsahuje i bezdrátové připojení Wifi a Bluetooth. Dále je možné dokoupit modul kamery, která se dá připojit přes plochý kabel. Rozmístění jednotlivých prvků lze vidět na obrázku 1.1.



Obrázek 1.2 Raspberry Pi 3 Model B [3]

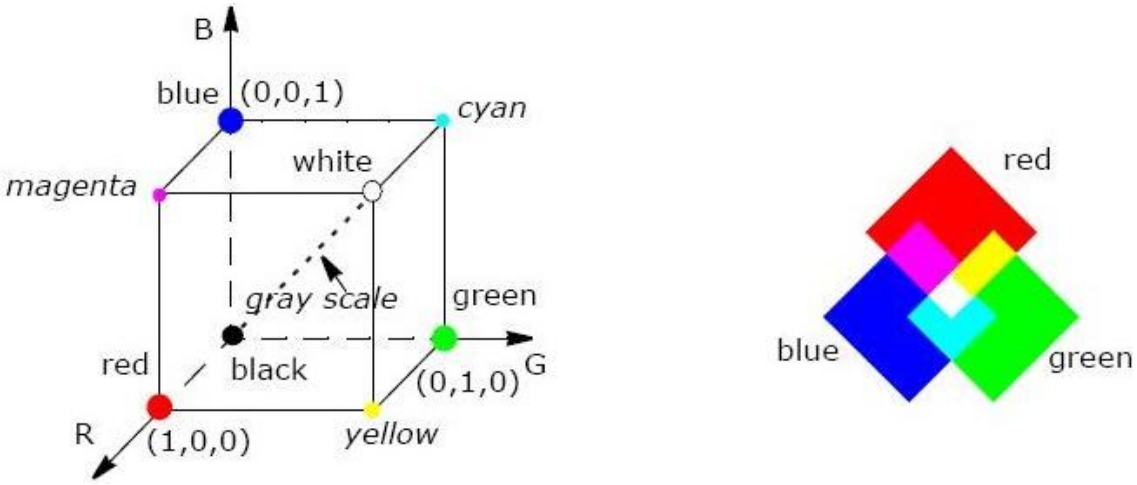
Základem je procesor s architekturou ARM, který zajišťuje veškerou funkcionalitu tohoto počítače. Doporučeným operačním systémem je Raspbian spadající pod systém Linux, ale může se na něj nainstalovat i jakýkoliv jiný systém kompatibilní s ARM platformou [4].

První Raspberry Pi byl vyvinut britskou nadací Raspberry Pi Foundation v roce 2012 za účelem podpory výuky informatiky [5].

2 DIGITÁLNÍ REPREZENTACE OBRAZU

2.1 Barevný model RGB

RGB neboli červená, zelená a modrá barva jsou základními barvami světelných paprsků. Jejich kombinací můžeme získat ostatní barvy z viditelného barevného spektra. Sloučením všech tří barev dostaneme barvu bílou. Jedná se o aditivní způsob míchání používaný v monitorech, projektorech a dalších zobrazovacích zařízeních [6]. Barevný model RGB a aditivní způsob míchání je zobrazeno na obrázku 2.1.



Obrázek 2.1 Barevný model RGB a aditivní způsob míchání [7]

Jeden takový barevný bod na obrazovce je digitálně reprezentován třemi binárními čísly. Obvykle se používá 8bitová hodnota a dekadicky může nabývat hodnot od 0 do 255. To je 256 hodnot pouze pro jednu barevnou složku. Počet možných kombinací je pak 256^3 což je ve výsledku celkem 16,7 mil. barev.

2.2 Jasová složka

Převodem ze tří matic s červenou, zelenou a modrou složkou dostaneme pouze jednu matici s jasovou neboli luminanční složkou. V podstatě se jedná o černobílý obraz, kde nejvyšší hodnota jasu je reprezentována jako bílá barva a nejnižší zase jako černá barva. Pro převod z barevného modelu RGB na černobílý obraz se využívá vztahu (1), kde proměnné R, G, B představují hodnoty od 0 do 255 [7].

$$Y = 0{,}299 \cdot R + 0{,}587 \cdot G + 0{,}114 \cdot B \quad (1)$$

Pro účely detekcí objektů v obraze se z většiny využívá pouze jasová složka, která je jednou ze tří složek barevného modelu YCbCr. Proto získat informace o ostatních složkách není potřeba.

2.3 Segmentace obrazu

Segmentace je nejdůležitějším procesem pro zpracování digitálního obrazu. Pomocí ní se obraz automaticky rozděluje na části pro znázornění různých objektů v obraze. Využívá různých metod a algoritmů pro její realizaci. Typickým cílem segmentace obrazu je identifikace popředí a určení oblastí v obraze odpovídajícím významnému prvku zachycené scény [8].

2.3.1 Prahování

Prahování je nejjednodušší metoda segmentace obrazu založená na hodnocení jasu každého pixelu [8]. Je založené na myšlence, že objekty a pozadí mají rozdílnou úroveň intenzity pixelu. Podle definované hodnoty prahu se každý pixel, který má menší hodnotu než tento práh, určí jako pixel pozadí a všechny ostatní pixely jsou považovány za pixely objektu. Výsledek prahování dostáváme po jediném průchodu obrazu [9].

2.3.2 Regionální metody

Regionální metody jsou metody, které jsou založeny na zjišťování podobnosti pixelů v nějaké vlastnosti. Touto vlastností může být například jas nebo statistické vlastnosti okolí pixelu [8].

Podstatou metod je, že konstrukce segmentu postupuje zdola nahoru, od jednoho pixelu po celý segment. Nejprve jsou nějakým algoritmem v obraze rozmístěny iniciální neboli semínkové pixely, obvykle rovnoměrně nebo náhodně, segment pak vzniká iterativním rozrůstáním se okolí iniciálního pixelu [8].

Při použití metod založených na růstu segmentu není zaručeno, že při různém počtu a rozmístění iniciálních pixelů bude výsledek segmentace identický. Na druhou stranu jsou tyto metody schopny segmentovat i takový obraz, který obsahuje značné množství šumu [8].

2.3.3 Metody založené na hranici

Jsou založeny především na detekci regionálních rozdílů ve vlastnostech obrazu. Patří k nim detekce hran nebo sledování hranice [8].

K detekci hran se obvykle používají gradientní operátory, např. Cannyho hranový detektor nebo Sobelův filtr. Protože výstupem gradientního operátoru je obraz, kde jsou sice hrany zvýrazněny, ale může obsahovat další artefakty odpovídající lokálním nehomogenitám v obraze, je třeba obraz dále upravit. K odstranění artefaktů vzniklých lokálními malými rozdíly obvykle postačuje prahování. Vzhledem ke svým vlastnostem mohou detektory hran vytvářet přerušované hranice i falešné hranice, obraz hranic je tedy třeba dále zpracovat [8].

Sledování hranice je postup aplikovatelný na obrazy obsahující především informaci o hranicích, například na výsledky použití gradientních filtrů (gradientní obraz). Cílem metod je v gradientním obraze právě jen skutečné hrany a vyloučit artefakty. Sledování hranice může selhávat u příliš zašumělých obrazů nebo u příliš komplikovaných tvarů segmentu [8].

2.3.4 Další metody

Aktivní kontura je pokročilá metoda segmentace obrazu. Metoda požaduje, aby jejím vstupem byla uzavřená křivka přibližně ohraničující segment, který je třeba ohraničit přesně. Na základě fyzikálních analogií se pak definují energie a síly, které v konečném důsledku deformují uzavřenou křivku tak, že se stane hranicí segmentu [8].

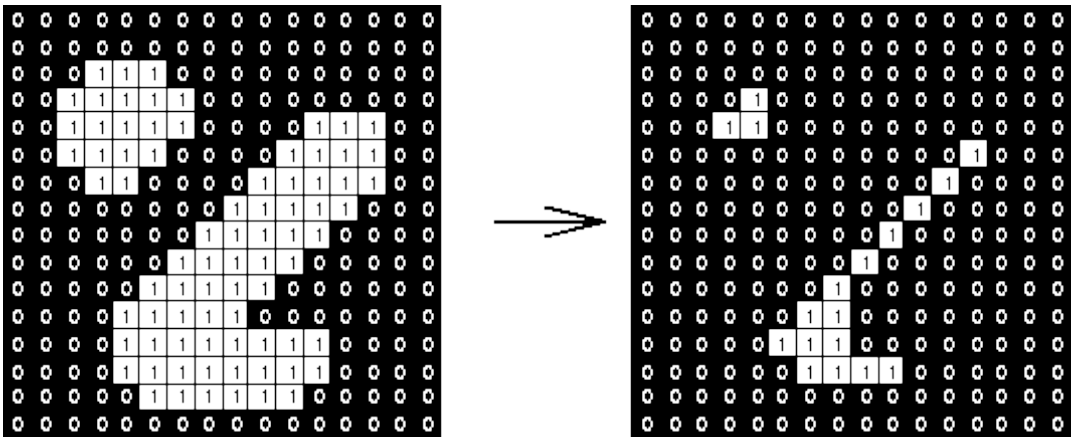
Dále metoda segmentace rozvodím je metoda založená na názorné představě zaplavování obrazu vodou. Jas pixelu je přitom chápán jako nadmořská výška. Jeden segment je pak vlastně oblastí vyplněnou vodou, hranice segmentů je místem, kde se stýkají dvě zaplavené oblasti oddělené souvislou skupinou pixelů s vyšší hodnotou jasu [8].

2.4 Matematická morfologie

Matematická morfologie se hlavně využívá u binárních obrazů, kde jsou pouze dvě barvy a to bílá nebo černá. Slouží k předzpracování obrazu jako je odstranění šumu, zjednodušení nebo zdůraznění objektů. Mezi dvě základní operace patří eroze a dilatace [10].

2.4.1 Eroze

Eroze skládá dvě bodové množiny s využitím vektorového rozdílu. Slouží pro zjednodušení struktury objektů. Objekty jednotkové tloušťky (relativní jednotka) zmizí a složité objekty spojené čárami jednotkové tloušťky se rozloží na několik jednodušších objektů [10]. Na obrázku 2.2 je zobrazena eroze s maskou 3x3 bodů. Procházejí se jednotlivé pixely obrázku. Střed masky se umístí na procházený pixel. Pokud maska nesouhlasí s aktuální oblastí jsou všechny body v oblasti nastaveny na nula [7].

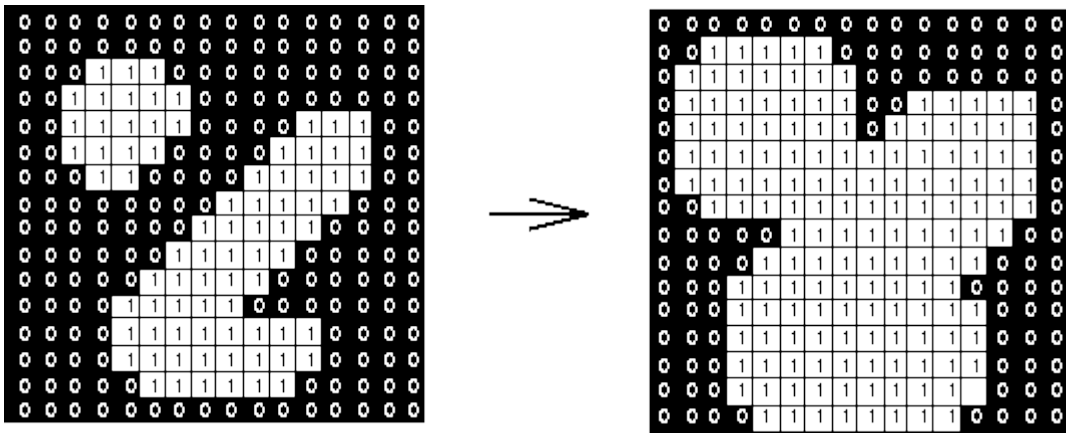


Obrázek 2.2 Eroze [11]

2.4.2 Dilatace

Dilatace se používá k zaplnění děr popř. zálivů. Skládá body dvou množin pomocí vektorového součtu [10]. Na obrázku 2.3 je zobrazena dilatace s maskou 3x3 bodů. Postup je stejný jako u eroze s tím rozdílem, že pokud maska nesouhlasí s aktuální

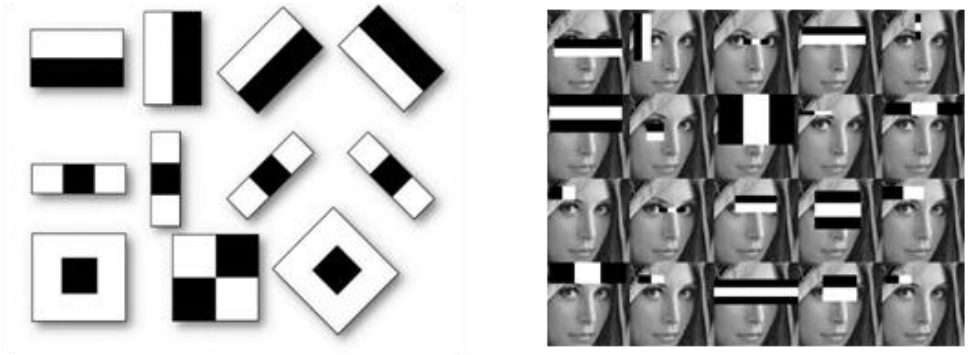
oblastí jsou všechny body v oblasti nastaveny na jedna [7].



Obrázek 2.3 Dilatace [11]

2.5 Haarovy příznaky

Slouží pro detekci objektů v obraze jako jsou obličeje, chodci, a jiné objekty. Využívají sumu hodnot jasů pixelů v jednotlivých oblastech. Například pro detekci obličeje je možné využívat vlastností, že jednotlivé přechody (čelo-oči, tváře-nos apod.) jsou detekovány ve stejném místě a mají podobné diference [12]. Systém je vybaven několika počty pozitivních obrazů jako jsou tváře různých osob na různých pozadích a negativními obrazy, na kterých nejsou žádné tváře, ale jiné objekty jako stůl, strom, dům apod.



Obrázek 2.4 Haarovy příznaky a jejich použití [14][15]

Z obrázku 2.4 je patrné využití jednotlivých přechodů jasů v obličeji pro jeho detekci.

Haarovy příznaky jsou tedy odvozeny od několika obdélníků, které se dělí podle typu informace, která má být s jejich pomocí detekována. Používají se hranové příznaky, čárové příznaky a příznaky středové. Bílá obdélníková oblast má stanovenu váhu a váha černé obdélníkové oblasti je vypočtena jako podíl ploch bílé a černé oblasti [13].

Při procesu generování příznaků jsou nejprve nastaveny nejmenší možné rozměry

příznaku a poté je příznak iterativně posouván oknem vždy po jednom pixelu buď ve vertikálním, nebo v horizontálním směru. Při každém posuvu je příznak přidán do seznamu všech příznaků. Pokud je příznak posunut přes celé okno, je zvětšena jeho velikost a opět je posouván oknem. Tento proces se opakuje, dokud není velikost příznaku větší než velikost okna [13].

3 KNIHOVNY OPENCV

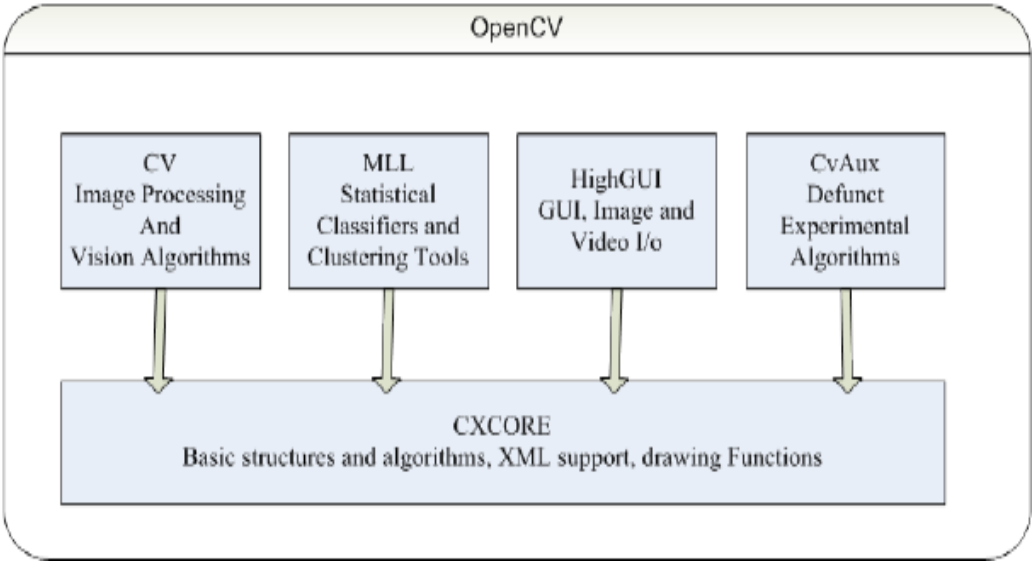
OpenCV je volně šiřitelná multiplatformní knihovna obsahující širokou škálu funkcí pro zpracování obrazu z oblasti počítačového vidění. Je napsána v programovacích jazycích C, C++ a lze ji provozovat na mnoha operačních systémech. Obsahuje nástroje pro analýzu obrazu jako je detekce objektů, analýza pohybu apod. Knihovna začala být vyvíjena v roce 1999 společností Intel a neustále se na ní pracuje [16].

3.1 Struktura

OpenCV se dělí do pěti částí. Základem je knihovna CxCore v níž jsou definovány základní datové typy a jejich funkce. Ostatní knihovny se na tuto odkazují.

CV knihovna obsahuje různé typy zpracování obrazu. V CvAux jsou experimentální funkce a algoritmy pro rozpoznávání tváře z obrazu. MLL je knihovna pro počítačové učení. HighGui slouží především pro řešení I/O operací včetně funkcí pro zpracování videa [17]. Schéma struktury knihoven je na obrázku 3.1.

Pod tyto knihovny spadá několik dalších hlavičkových souborů. Pro tuto práci je například důležitá knihovna objdetect obsahující funkce pro detekci objektů v obraze [12].



Obrázek 3.1 Struktura knihovny OpenCV [18]

4 VYUŽITÍ KNIHOVEN OPENCV

4.1 Načtení a zobrazení vstupních dat

OpenCV nabízí několik nástrojů pro načtení obrazových souborů. Tyto nástroje jsou součástí knihovny HighGUI [19].

Pomocí algoritmu uvedeného níže se načítají a zobrazují jednotlivé snímky z kamery, které je možno zpracovávat pod řádkem `if( !frame ) break;` kontrolující jestli se snímky načítají.

```
#include <highgui.h>
#include <cv.h>
#include <stdio.h>

int main( int argc, char** argv ) {
    IplImage* frame = 0;
    cvNamedWindow( "Camera", CV_WINDOW_AUTOSIZE );
    CvCapture* capture = 0;
    capture = cvCaptureFromCAM( CV_CAP_ANY );
    while(1) {
        frame = cvQueryFrame( capture );
        if( !frame ) break;
        cvShowImage( "Camera", frame );
        char c = cvWaitKey(33);
        }
    cvReleaseCapture( &capture );
    cvDestroyWindow( "Camera" );
}
```

4.2 Zpracování obrazu

V knihovně cv se dá využít funkcí pro různé barevné transformace obrazu. Pro změnu obrazu z RGB na černobílý obraz se zavolá funkce cvtColor() s parametry o vstupním obrazu, výstupním obrazu a typu převodu [20].

```
cvtColor (image, gray_image, CV_BGR2GRAY);
```

Dalším zpracováním se dá považovat převod na binární obraz. K tomu slouží funkce threshold(). První dva parametry jsou znovu vstupní a výstupní obraz. Ostatní tři jsou hodnota od které má dojít k prahování, maximální hodnota (obvykle 255), a typ prahování [20].

```
threshold(image, image_threshold, value, max_BINARY, type);
```

Funkce pro morfologické operace se volají zapsáním erode() nebo dilate(). Jejich třetím parametrem je informace o vlastnostech masky. Pomocí funkce getStructuringElement() se zapíše typ masky, velikost a výchozí bod. Na výběr jsou tři typy masky MORPH_RECT (obdélník), MORPH_ELLIPSE (elipsa) a

MORPH_CROSS (tvar kříže). Zápisem Point(-1,-1) se nastaví maska na střed testovaného bodu [20].

```
Mat element = getStructuringElement(type, Size(x,y), Point(-1,-1) );
erode( image, image_erosion, element );
dilate( image, image_dilatation, element );
```

Další úpravou může být zmenšení velikosti původního obrázku. Dá se tím výrazně zrychlit zpracování, jelikož se prochází menší počet obrazových bodů [12].

```
resize(image,small_image,Size(cols,rows));
```

4.3 Jednoduché kreslení

Pro vyznačení docházených detekcí se dají použít funkce pro kreslení obdélníku nebo kruhu.

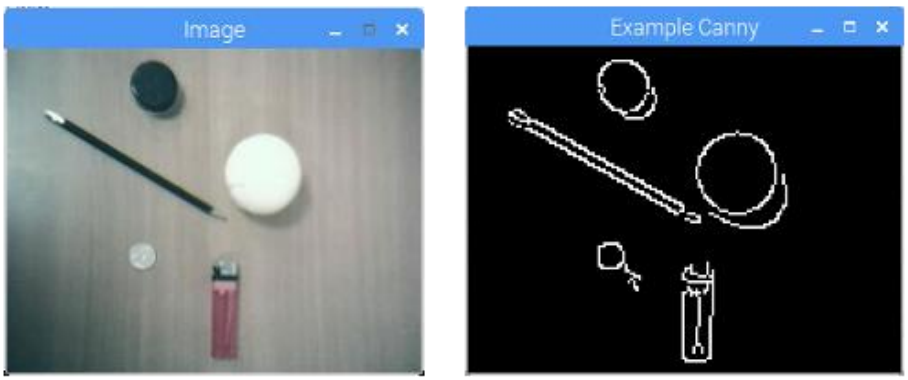
```
void rectangle(Mat& img, Point pt1, Point pt2, const Scalar& color,
int thickness=1, int lineType=8, int shift=0)

void circle(Mat& img, Point center, int radius, const Scalar& color,
int thickness=1, int lineType=8, int shift=0)
```

4.4 Příklady detekcí

4.4.1 Hranový detektor Canny

Detekce hran je založená na procházení jasového obrazu a určení oblastí kde se prudce změní hodnota jasu. Tyto oblasti jsou pak vyhodnoceny jako hrana [21]. Mezi nejpoužívanější detektory patří Cannyho operátor. Dalšími jsou například operátory Roberts, Prewitt nebo Sobel. Příklad výsledku zpracování Cannyho detektoru hran je na obrázku 4.1.



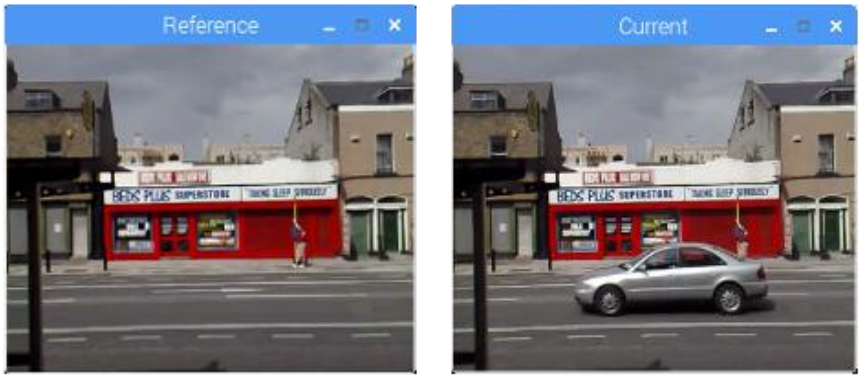
Obrázek 4.1 Cannyho detektor hran [22]

Tento detektor je považován za ideální, protože splňuje určité podmínky pro ideální detekci hran. Mezi tyto podmínky patří omezení detekování falešných hran,

minimální vzdálenost mezi skutečnou a detekovanou hranou a detekování každé hrany pouze jednou. Pro dosažení požadovaných vlastností je zapotřebí aplikovat jisté postupy [9]. Prvním krokem je eliminace šumu Gausovským filtrem. Dále určení velikosti a směru gradientu, který je důležitý pro další kroky. Jedná se standardní detekci hran, kde je využit například Sobelův operátor. Poté se vybírá z hodnot gradientů lokální maxima. Tím se odeberou body, které jsou v okolí proti směru gradientu nižší a zajistí se ztenčení detekované hrany. Posledním krokem se metodou prahování odeberou nevýznamné hrany [23].

4.4.2 Detekce pohybu

Metoda spočívá v porovnávání aktuálního snímku na kameře a referenčního, na kterém není žádný pohybující se objekt [24]. To znázorňuje obrázek 4.2.

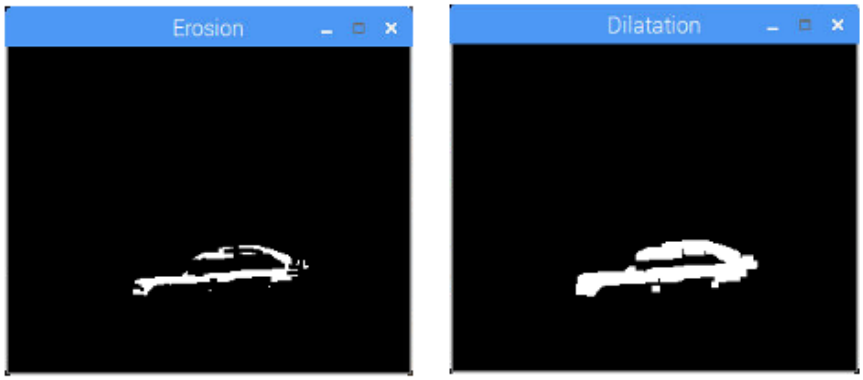


Obrázek 4.2 Referenční a aktuální snímek [25]

Snímky se převedou na černobílý obraz a mezi sebou se odečtou. Výsledkem jsou světlé části reprezentující změnu. Může nastat situace, že při odečtení dvou bodů bude výsledkem záporná hodnota. Proto je nutné počítat s absolutní hodnotou. Dále se výsledek metodou prahování převede na binární obraz viz obrázek 4.3.

Obrázek 4.3 Rozdíl snímků a aplikovaná metoda prahování

Kvůli případnému šumu v podobě bílých bodů mino detekovaný objekt je potřeba tento šum odstranit. K tomu poslouží morfologické operace. Provede se eroze a následně dilatace na erozi pro vyplnění objektu viz obrázek 4.4.



Obrázek 4.4 Eroze prahovaného snímku a dilatace erodovaného snímku

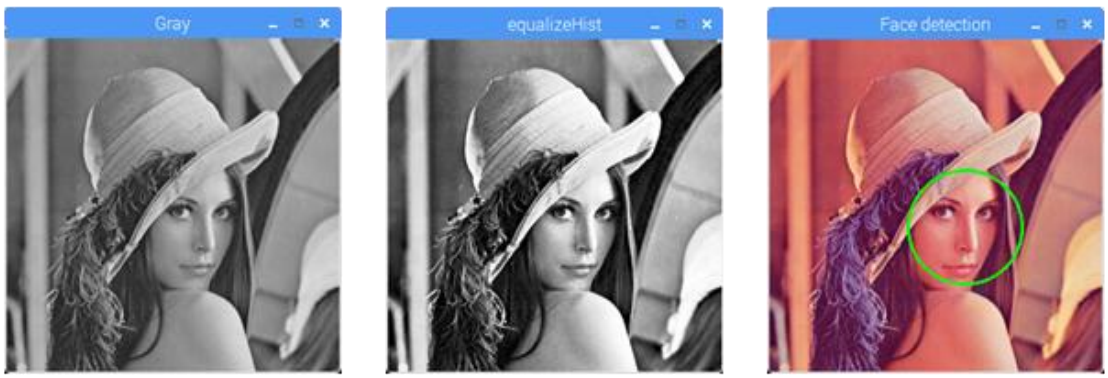
Dilatace je posledním zpracováním a výsledný obraz značí detekovaný pohyb.

4.4.3 Detekce obličeje

Metoda funguje na základě kaskádní detekce obličejů pomocí Haarových příznaků. Načte se trénovací skupina obrazů a dále se procházejí jednotlivé oblasti ve snímku. OpenCV obsahuje několik natrénovaných dat zapsaných v souborech typu xml. Na výběr jsou například soubory pro detekci očí, částí těla apod. [12]. V tomto případě se načte soubor *haar_frontalface_alt.xml*. Nejdříve je ovšem potřeba převést obraz na černobílý a dále ještě na ekvalizovaný pomocí funkce *equalizeHist()* viz obrázek 4.5.

```
equalizeHist(image_gray,image_equalize);
```

Základní vlastností ekvalizovaného obrazu je rovnoměrné zastoupení jasových úrovní. Zjednodušeně řečeno je ve výstupním obrazu věnováno více jasového prostoru objektům, které zabírají větší plochu, než objektům menším [26].



Obrázek 4.5 Černobílý obraz, ekvalizovaný obraz a výsledek detekce obličeje [27]

4.5 Ukládání výstupních dat

Pokud budeme chtít udělat záznam detekce, nejspíš ho zaznamenáme do obrazového souboru. K tomu můžeme využít například funkci pro zápis *imwrite()*.

```
imwrite( "/home/pi/image_out.jpg", image_out );
```

5 PŘÍSTUP K ZÁZNAMŮM DETEKCE A OVLÁDÁNÍ SYSTÉMU

Jelikož nebude k systému připojen žádný monitor, klávesnice a myš, bude potřeba získat přístup přes internet pro ovládání systému. Jednotlivé záznamy se budou na zařízení ukládat do určitého adresáře a následně načítat na webovou stránku za pomocí PHP souboru. Informace o snímcích, jako je datum pořízení, se budou ukládat do databáze MySQL. K realizaci bude potřeba zřídit na systému webový server aby stránka mohla být na síti dostupná. Přímo pro Raspberry Pi je možné nainstalovat webový server zvaný Apache. Obsahem PHP souboru můžou být kromě PHP kódu i kódy jiných jazyků jako jsou HTML, CSS nebo Javascript.

5.1 HTML a CSS

HTML je hlavním z jazyků používaného pro tvorbu webových stránek, které jsou propojeny hypertextovými odkazy, umožňující publikaci dokumentů na Internetu [28].

Jazyk HTML je charakterizován množinou značek neboli tzv. tagů a jejich vlastností neboli atributů definovaných pro danou verzi. Mezi značky se uzavírají části textu dokumentu a tím se určuje význam obsaženého textu. Názvy jednotlivých značek a jejich vlastností se uzavírají mezi úhlové závorky. Část dokumentu tvořená otevírací značkou, nějakým obsahem a odpovídající ukončovací značkou tvoří prvek neboli element dokumentu [28]. Ukončovací značka je navíc tvořena lomítkem. Prvky ale nemusí vždy mít ukončovací značku. To záleží na tom, jestli je značka párová nebo nepárová. Například pro hlavní nadpis se může použít tag h1. Prvek obsahující hlavní nadpis je uveden níže.

```
<h1>Raspberry Pi Camera</h1>
```

Dále pro vložení odkazu se využívá tag a. Pro vložení obrázku zase tag img, který je jedním z nepárových tagů.

```
<a href="url">text odkazu</a>
<img src="obrázek.jpg">
```

Existuje mnoho dalších tagů, které mají své určité funkce pro webové stránky. Ještě je vhodné se zmínit o tagu div, který slouží jako oddělovač části stránky. Zahrnuje v sobě libovolně velkou oblast textu včetně nadpisů, obrázků a tabulek [29]. Nejvíce se využívá pro určování pozic částí stránky, aby celý dokument byl systematicky rozdělen a byl tak úsporně využit prostor stránky. K tomu musí být oddělovače definované pomocí CSS.

CSS jsou zkratkou pro kaskádové styly (Cascading Style Sheets) a jsou v informatice jazykem pro popis způsobu zobrazení elementů na stránkách napsaných v jazycích HTML, XHTML nebo XML [30].

Definice kaskádových stylů sestává z několika pravidel. Každé pravidlo obsahuje selektor a blok deklarací. Každý blok deklarací pak obsahuje deklarace oddělené

středníky a každá deklarace sestává z identifikátoru vlastnosti, následuje dvojtečka a hodnota vlastnosti [30]. Pro zápis kaskádových stylů slouží v HTML dokumentu párový tag style. Níže je uveden příklad zápisu.

```
<style type="text/css">
a{    color:#F00;
      font-family:Calibri;
}
</style>
```

Z příkladu je patrné, že každý odkaz na stránce bude mít červenou barvu a font Calibri. Pro určení vlastností oddělovače vypadá zápis například tak jak je uvedeno níže.

```
div#box{
      color:#00F;
      width:1000px;
      margin-top:20px;
}
```

V selektoru se za křížkem zapíše název oddělovače a pak blok deklarací ve složených závorkách. Podle příkladu bude v tomto oddělovači modrý text, jeho šířka 1000 pixelů a odsazení od horní části 20 pixelů. Zápis oddělovače v HTML kódu vypadá následovně.

```
<div id="box">
      obsah
</div>
```

5.2 PHP

PHP je skriptovací programovací jazyk. Je určený především pro programování dynamických internetových stránek a webových aplikací například ve formátu HTML, XHTML či WML. PHP lze použít i k tvorbě konzolových a desktopových aplikací. Pro desktopové použití existuje kompilovaná forma jazyka [31].

Nejčastějším prvkem webové stránky pro ovládání systému jsou tlačítka, přepínače, výběrová a zadávací pole, které se na internetových stránkách nejvíce používají k vyplnění nějakého formuláře. Zde jsou použité pouze pro ovládání programu přes textové soubory. Například pro zapnutí nebo vypnutí zpracovávání snímku k detekci se zvolí přepínač na ON nebo OFF a potvrzením tlačítka se otevře textový soubor a zapíše jednička pro zapnutí nebo nula pro vypnutí. V programu se v každém cyklu soubor otevře a za pomocí podmínky se zjišťuje, jestli má zpracování proběhnou či nikoli. Tento způsob je využit i u dalších parametrů. Níže je uveden kód v PHP.

```
<?php
if (isset($_REQUEST["apply"]) and ($_REQUEST["apply"]=="APPLY"))

{     //on/off detection (write to txt 1 or 0)
      if ($_REQUEST['move'] == 'A')
      {
                  $file = fopen("data/on-off.txt", "w");
                  fwrite($file, 1);
```

```
                fclose($file);

        }
        if ($_REQUEST['move'] == 'B')
        {
                $file = fopen("data/on-off.txt", "w");
                fwrite($file, 0);
                fclose($file);
        }
}
?>
```

Tyto řádky pouze zajišťují co se má udělat při stisknutí tlačítka. Pro zobrazení tlačítka a přepínače se musí přidat kód v HTML uvedený níže.

```
<form action="index.php" method="get" id="index">

<label for="move">Move detection</label><br/>

<input name="move" type="radio" value="A" checked="checked"/>ON
<input name="move" type="radio" value="B"/>OFF

<input class="apply" name="apply" type="submit" value="APPLY" />

</form>
```

V programu C++ se pak pro čtení ze souboru používá kód uvedený níže, který otevře soubor, načte první řádek a obsah uloží do proměnné se kterou se dále pracuje po zavření souboru.

```
FILE *ptrfile;
ptrfile = fopen("/var/www/html/data/on-off.txt","r");
if (ptrfile!=NULL)
fscanf(ptrfile, "%d", &on-off);
fclose(ptrfile);
```

5.3 JavaScript

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk. Hlavně se používá jako interpretovaný programovací jazyk pro WWW stránky, často vkládaný přímo do HTML kódu stránky. Jsou jím obvykle ovládány různé interaktivní prvky GUI (tlačítka, textová políčka) nebo tvořeny animace a efekty obrázků. Jeho zápis zdrojového textu patří do rodiny jazyků C/C++/Java. Ale JavaScript je od těchto jazyků zásadně odlišný, sémanticky (vnitřně) jde o jiný jazyk [32].

Pro webovou stránku pro přístup je zde JavaScript použitý pro automatické obnovení části stránky jako je zobrazení času a živý přenos z kamery. Část stránky je oddělena pomocí oddělovače. Níže je uveden kód JavaScriptu.

```
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.3/jquery.min.js"></
script>
<script type="text/javascript">
```

```
  setInterval("$('#info').load('info.php');", 1000);
</script>
```

Funkcí *setInterval* se nastaví doba intervalu pro obnovení v milisekundách a za křížkem se zapíše příslušný název oddělovače ve kterém je zahrnut PHP soubor s kódem, který se má automaticky aktualizovat jak je uvedeno v kódu výše. Musí být tedy vytvořen druhý PHP soubor. Kód pro zahrnutí druhého souboru do oddělovače v hlavním souboru je uveden níže.

```
<div id="info">
<?php
include ("info.php");
?>
</div>
```

5.4 MySQL a phpMyAdmin

Programový systém phpMyAdmin je nástroj napsaný v jazyce PHP umožňující jednoduchou správu obsahu databáze MySQL prostřednictvím webového rozhraní. V současné době umožňuje vytvářet/rušit databáze, vytvářet/upravovat/rušit tabulky, provádět SQL příkazy a spravovat klíče. Jedná se o jeden z nejpopulárnějších nástrojů pro správu databáze. Je k dispozici v 57 jazycích [33].

Za pomocí programu v C++ se do databáze ukládají informace o uložených záznamech a pomocí nich se zase zobrazují na webové stránce. Kód pro přidávání záznamů do tabulky v databázi je uveden níže.

```
pConnection = mysql_init(NULL);

mysql_real_connect(pConnection,"localhost","root","password","pi",0,NU
LL,0);

sprintf(Query, "INSERT into pirecords (idday,date,time,name,dtype)
values ('%d','%s','%s','%s','%s')", idday,date,time,name,dtype);

mysql_query(pConnection, Query);
```

Tabulka se záznamy má 5 sloupců s informacemi o celkovém číslu záznamu, číslu záznamu pro aktuální den, datum a čas pořízení, dále název souboru snímku a typ detekce (pohyb nebo Haarovy znaky).

PHP kód pro spojení s databází MySQL vypadá následovně.

```
$connection = mysql_connect("localhost", "root", "password");

if (!$connection){
      die('Could not connect: ' . mysql_error());
      }
      else
      {
      mysql_select_db("pi");

      }
```

Pokud nebude navázáno spojení s MySQL, vypíše se chybové hlášení a v opačném případě se vybere databáze, která se bude využívat. Dále se dá určit, co se s databází bude dělat jako například získat názvy snímků pro jejich načtení na stránku. PHP kód pro zobrazení snímků na webové stránce vypadá následovně.

```
$query="";
$query=$query." SELECT ";
$query=$query." a.idday,a.timex,a.day, a.name, a.dtype ";
$query=$query." FROM ";
$query=$query." pirecords AS a ";

$result = mysql_query($query);

while ($record = mysql_fetch_array($result))
{
print("<img src=\"det/".$record["name"]."_".$record["idday"].".jpg\"
alt=\"".$record["name"]."\"/>");

print($record["day"]." | ".$record["time"]);
print($record["idday"]." | ".$record["dtype"]);
}
```

Do proměnné query se uloží SQL příkaz pro vypsání záznamů z tabulky v databázi. Ten se následně odešle do SQL ke zpracování a cyklus while poté načítá jednotlivé řádky z vybraných sloupců uložené v proměnné record. V každém cyklu se tedy načte obrázek a jeho informace o datu pořízení a typu detekce.

5.5 Zabezpečení stránky

Webové stránky se dají zabezpečit pomocí souborů .htaccess a .htpasswd, které se vloží na webový server k PHP souboru index.php.

Obsahem souboru .htaccess je cesta k tomuto souboru a v .htpasswd jsou zapsány jména uživatelů a hesla. Obsah .htaccess může vypadat následovně.

```
AuthUserFile  /var/www/html/.htpasswd
AuthName "Please login"
AuthType Basic
Require valid-user
```

A obsah .htpasswd

```
user:password
```

Bohužel v Rapsberry Pi nastal problém s povolením těchto souborů na webovém serveru, který se nepodařilo vyřešit. Místo toho se při zadání adresy stránky zobrazí pouze dvě textová pole pro zadání loginu a hesla a tlačítko pro potvrzení. Při správném zadání se přejde na hlavní stránku. Zadávání je řešeno stejným způsobem jako nastavení parametrů pro zpracování.

5.6 Odstranění záznamů

Odstranění záznamů probíhá přes program v C++. Kód pro odstranění konkrétního souboru je uveden níže.

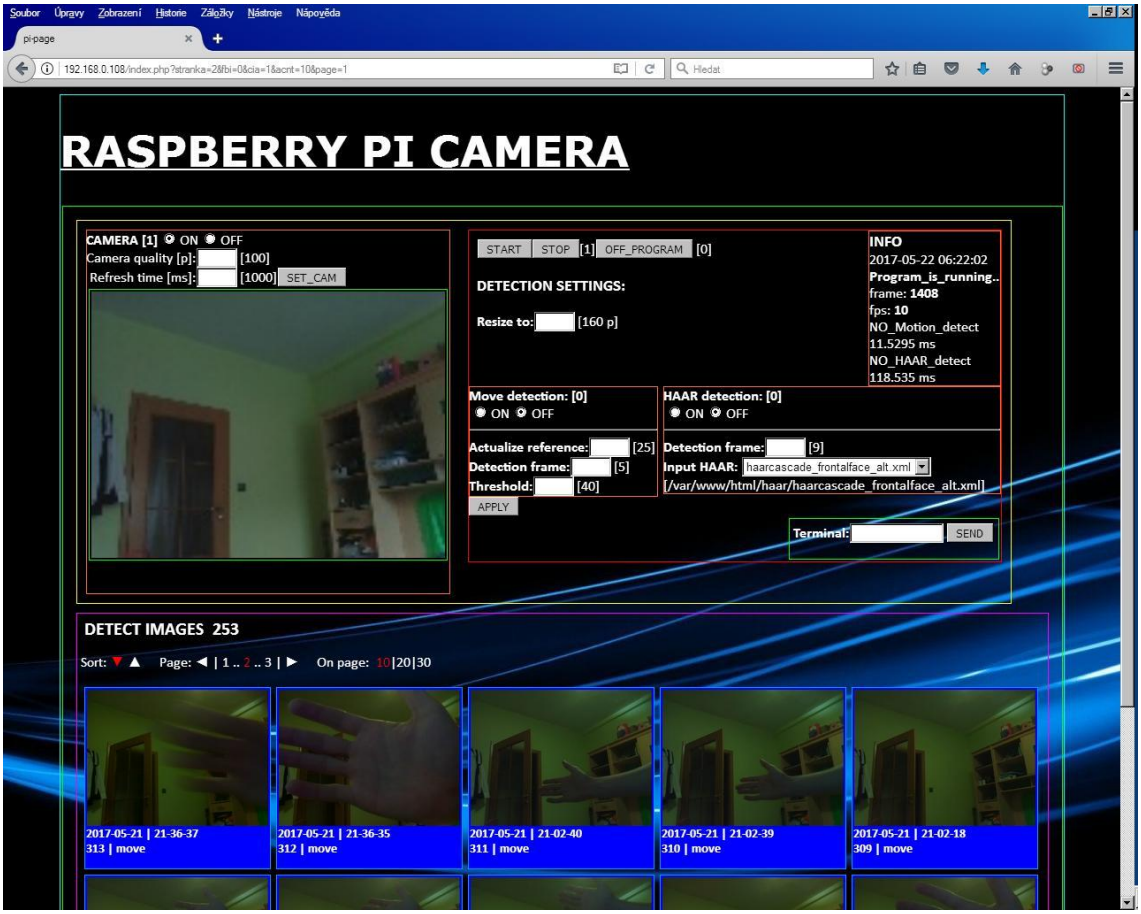
```
#include <stdio.h>

int main ()
{
  if( remove( "path/to/file.jpg" ) != 0 )
    perror( "Error deleting file" );
  else
    puts( "File successfully deleted" );
  return 0;
}
```

Dále se musí odstranit i informace o snímku uložené v databázi. To zajišťuje stejný kód jako při vkládání záznamů do tabulky v databázi s tím rozdílem, že místo SQL příkazu INSERT se použije DELETE.

5.7 Vzhled webové stránky

Webová stránka pro ovládání systému obsahuje živí přenos z kamery, nastavení parametrů pro detekci a možnost procházení záznamů. Její zobrazení je obrázku 5.1.



Obrázek 5.1 Zobrazení webové stránky pro ovládání systému

Na levé straně se zobrazuje živý přenos z kamery. Je možné nastavit počet řádků snímku pro zobrazení a doba intervalu pro automatické obnovení snímku. Zobrazení se dá i vypnout nebo zapnout. Pokud program běží dá se načítání snímků zapnout nebo vypnout pomocí tlačítek START a STOP na pravé straně stránky, a nebo program úplně vypnout tlačítkem OFF_PROGRAM.

Dále je zde možné nastavit parametry pro zpracování snímku. Zapnutí nebo vypnutí detekce pohybu nebo detekce pomocí Haarových znaků. U detekce pohybu se nastavuje po kolika snímcích se má aktualizovat referenční snímek, po kolika snímcích má dojít ke zpracování a hodnota prahu. U detekce pomocí Haarových znaků se nastavuje po kolika snímcích má dojít ke zpracování a vybírá se XML soubor s natrénovanými daty. Tyto parametry se potvrzují stisknutím tlačítka APPLY. U každého parametru se v hranatých závorkách vypíše obsah textového souboru, což slouží pouze pro informaci. Je zde ještě jedna část, která se automaticky aktualizuje, jelikož obsahuje aktuální čas, počet načtených snímků od začátku programu, počet načtených snímků za sekundu a informace o tom, jestli dochází nebo nedochází k detekování a o době zpracování.

Ve spodní části se zobrazují detekované snímky s informacemi o datu a času pořízení a je možné nastavit kolik snímků se má zobrazit na stránku a pak se přepínat mezi stránkami a dále zvolit jestli se snímky mají načítat vzestupně, nebo sestupně. Ve výchozím nastavení je sestupné řazení, aby se záznamy zobrazovali od nejnovějšího po

nejstarší.

5.8 Běh programu

Při spuštění Rapberry Pi se automaticky spustí program. To je řešeno přepsáním některých konfiguračních souborů. Dále je program v tvz. režimu čekání, kdy se může zadat velikost snímku ke zpracování. Přesněji je v prvním cyklu while dokud není v textovém souboru pro start zapsána jednička. Při stlačení tlačítka START se program přepne do tvz. režimu běh programu. Pro přesnost je v druhém cyklu while dokud není v textovém souboru pro start zapsána nula. Zde se načítají snímky z kamery a může se zapínat nebo vypínat detekce pohybu nebo detekce pomocí haarových znaků . Znova se dá mezi těmito režimy přepínat. Tlačítkem OFF_PROGRAM na webové stránce se celý program vypne pokud je v režimu čekání. Znova se dá pak zapnout pouze přes vzdálenou plochu nebo odpojením a zapojením napájecího napětí.

6 TESTOVÁNÍ SYSTÉMU

Systém se testoval u detekce pohybu a detekce pomocí haarových znaků. Testy byly zaměřeny na nastavení parametrů systému a při různých situacích.

6.1 Detekce pohybu

Detekce pohybu se testovala z pohledu rozlišení snímku, hodnoty prahu a nastavení rozestupu snímků mezi zpracováním a aktualizováním referenčního snímku.

6.1.1 Rozlišení snímku

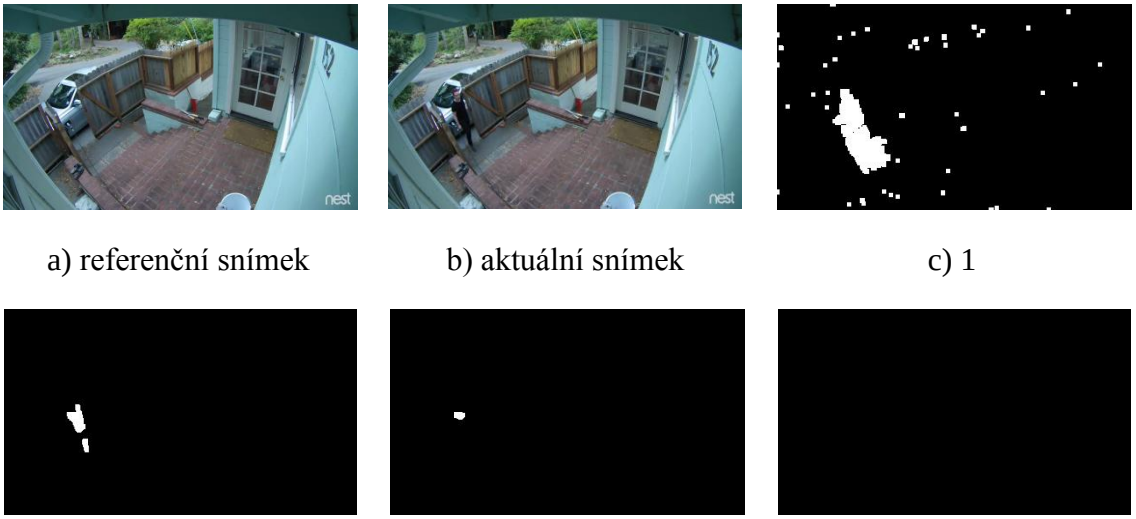
Velikost snímku má vliv pouze na čas zpracování. Od změny snímku na černobílí až po dilataci se čas zpracování zvyšuje s vyšším rozlišením snímku. Výsledky měření jsou uvedeny v tabulce 6.1.

Tabulka 6.1 Závislost počtu řádků ve snímku na době zpracování

Počet řádků ve snímku [p]	Doba zpracování [ms]
60	4
160	11
200	17
240	25
360	60
480	100
720	240
1080	530

6.1.2 Hodnota prahu

Tento test slouží k rozhodnutí nastavení ideální hodnoty prahu. Na prvních dvou dílčích snímcích z obrázku 6.1 je zobrazen referenční a aktuální snímek a na obrázcích 6.1c až 6.1f výsledný dilatovaný snímek s uvedením aplikované hodnotou prahu.

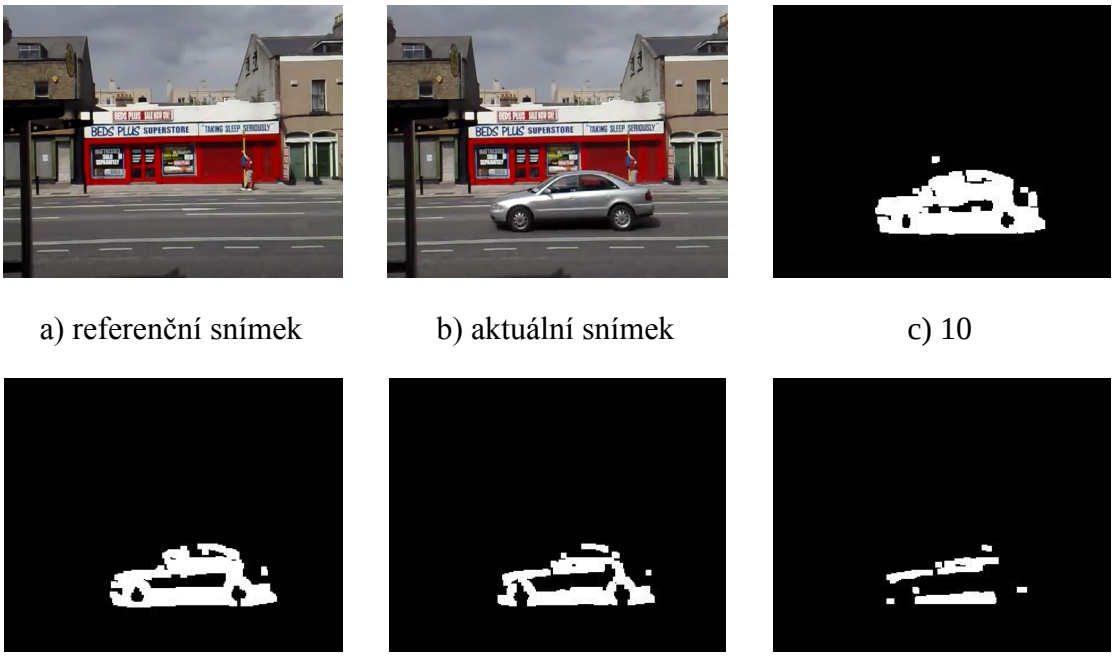


a) referenční snímek b) aktuální snímek c) 1

d) 50 e) 100 f) 150

Obrázek 6.1 Test hodnoty prahu č.1 [34]

Jak jde vidět při hodnotě prahu 1 se vyskytl šum a naopak při hodnotě 150 se nevyskytl žádný bílý bod. Proto se netestovaly vyšší hodnoty. Při hodnotě 100 byla vytvořena menší bílá plocha než jaký jde vidět rozdíl mezi aktuálním a referenčním snímkem. Hodnota 50 je v tomto testu nejlepší, jelikož je bílá plocha srovnatelná s pozorovatelným rozdílem. Další test je tedy zaměřen na hodnoty mezi 10 až 75.



a) referenční snímek b) aktuální snímek c) 10

d) 25 e) 50 f) 75

Obrázek 6.2 Test hodnoty prahu č.2 [25]

Na dílčím obrázku 6.2c je ještě pozorovatelná malá bílá plocha mimo objekt a na

obrázku 6.2f je zase objekt málo vyplněný. Z toho se dá usoudit, že ideální hodnota prahu je mezi 25 až 50.

6.1.3 Nastavení rozestupu snímků

Kromě hodnoty prahu se na webové stránce u detekce pohybu nastavuje po kolika načtených snímcích z kamery má dojít ke zpracování a k aktualizování referenčního snímku. Proto se provedlo pozorování aktuálních snímků z kamery, referenčních snímků a dilatovaných snímků v závislosti na nastavených hodnotách. Výsledky jsou shrnuty v následujícím odstavci.

Může nastat situace, že v okamžiku zpracování snímku se aktualizuje referenční snímek. Tím pádem se do referenčního snímku uloží právě detekovaný objekt. Proto bude lepší když se nastaví nízký rozestup mezi zpracováním a aktualizováním snímku. Například 5 a 30. To znamená, že při každém šestém zpracování se v případě pohybu aktualizuje referenční snímek s objektem, což bude mít za následek falešnou detekci pohybu a uložení dalších minimálně 6 snímků, kde nemusí být zaznamenaný pohybující se objekt. Pokud by se tedy nastavil vysoký rozestup, například 5 a 500, zaznamená se takovýchto snímků 100. Kdyby se zase nastavil nízký rozestup nemusel by se detekovat pomalý pohyb, protože by rozdíly mezi snímky byly minimální.

6.1.4 Úspěšnost detekce

Systém se testoval na chodbě v budově FEKTu a byl spuštěn přibližně 1 hodinu. Nastaveno bylo po 5 snímcích zpracování, po 25 snímcích aktualizování referenčního snímku a velikost prahu 40. Zaznamenáno bylo 2065 snímků a z toho 31 snímků, kde byla prázdná chodba. Ve výsledku je to 1,5% snímků navíc. Při každém průchodu se kontrolovalo jestli se v informačním okně zobrazuje hláška o detekci pohybu a nestalo se, že by systém pohyb nezaznamenal. Na obrázku 6.3 je zobrazen záznam snímků při jednom průchodu osob.

a) 1. záznam b) 2. záznam c) 3. záznam

d) 4. záznam e) 5. záznam f) 6. záznam

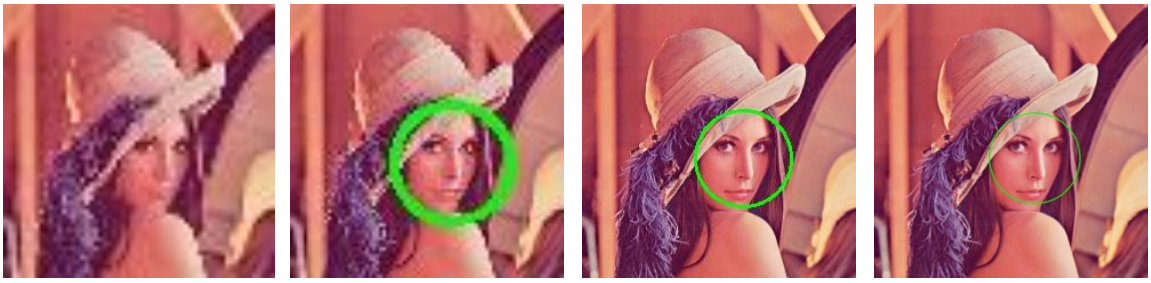
Obrázek 6.3 Záznam snímků při detekci pohybu

6.2 Detekce pomocí Haarových znaků - obličej

Detekce obličeje se testovala za pomocí souboru *haar_frontalface_alt.xml* uložený ve složkách OpenCV jako příkladový soubor pro detekci pomocí Haarových znaků. Všechny testované snímky nebo videa byly pořízeny z různých internetových zdrojů nebo z vlastních záznamů. Byly vybrány tak, aby splňovaly kritéria pro testování z různých hledisek, podmínek a situací. Například z hlediska vzdálenosti obličeje od kamery, úhlu natočení apod.

6.2.1 Rozlišení snímku

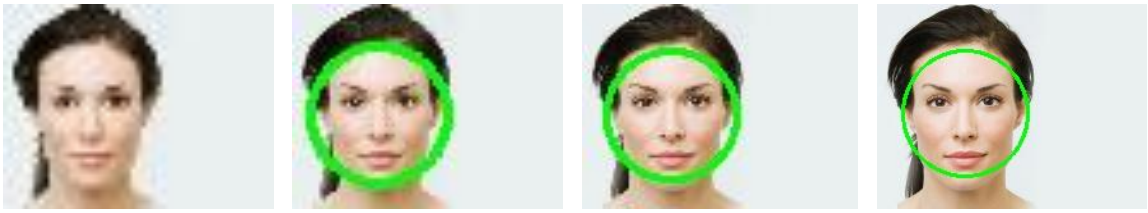
Byla otestována úspěšnost detekce na rozlišení obrázku a době zpracování. Pod dílčími obrázky je uveden počet řádků v obrázku a doba zpracování v milisekundách.



a) 60p - 3ms b) 80p - 8ms c) 240p - 206ms d) 720p - 2905ms

Obrázek 6.4 Detekce obličeje v závislosti na rozlišení snímku č.1 [27]

Na obrázku 6.4 lze vidět, že detekce byla úspěšná kromě výšky snímku 60p. Na dalším testu byla detekce úspěšná i při 60p viz obrázek 6.5.



a) 40p - 1ms b) 60p -10ms c) 80p - 25ms d) 160p - 103ms

Obrázek 6.5 Detekce obličeje v závislosti na rozlišení snímku č.2 [35]

Důvodem zřejmě bude, že když je obličej blíž ke kameře, může být detekce úspěšná i při nižší kvalitě. Proto by bylo vhodné otestovat rozlišení snímku na vzdálenosti obličeje od kamery. Nižší kvalita bude představovat větší vzdálenost.

a) 120p - 66ms b) 160p - 159ms c) 240p - 439ms

d) 120p - 47ms e) 160p - 120ms f) 240p - 342ms

Obrázek 6.6 Detekce obličeje v závislosti na rozlišení snímku č.3 [36][37]

Z obrázku 6.6 lze vidět, že s lepší kvalitou snímku se detekovali i vzdálenější obličeje. Testovali se ještě vzdálenější obličeje. Na dílčím obrázku 6.7a nebyla detekce úspěšná, ale na 6.7b se už obličeje detekovali kromě jednoho. Obrázky 6.7c a 6.7d znázorňují situaci na ulici. U 6.7d se detekovali tři pouze obličeje a doba zpracování byla navíc větší jak 2 sekundy. Na obrázku 6.7c došlo k falešné detekci, kde zřejmě byla nalezena podobnost s obličejem.

a) 240p - 444ms

b) 360p - 1004ms

c) 360p - 1189ms

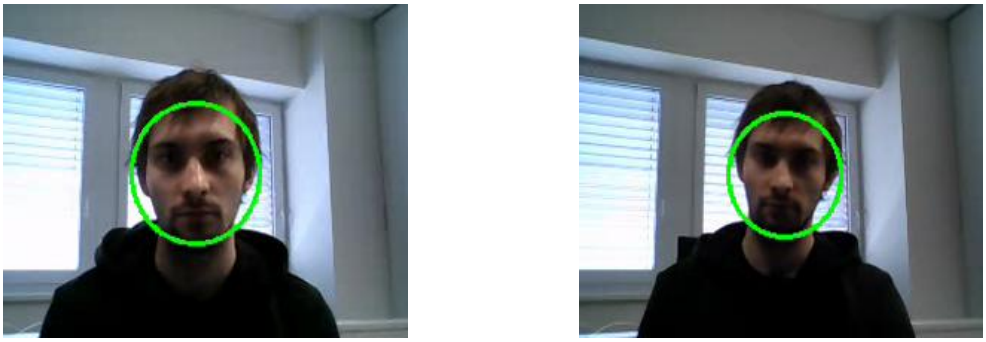
d) 480p - 2295ms

Obrázek 6.7 Detekce obličeje v závislosti na rozlišení snímku č.4 [38][39]

Ideální hodnota výšky snímku pro obličej přímo před kamerou je zhruba od 80p a pro obličej ve větší vzdálenosti asi od 160p. Od výšky snímku 360p je doba zpracování větší jak 1 sekunda. Aby se načítal alespoň jeden snímek za sekundu, musela by výška snímku ke zpracování být nižší jak 360p.

6.2.2 Úhel natočení obličeje

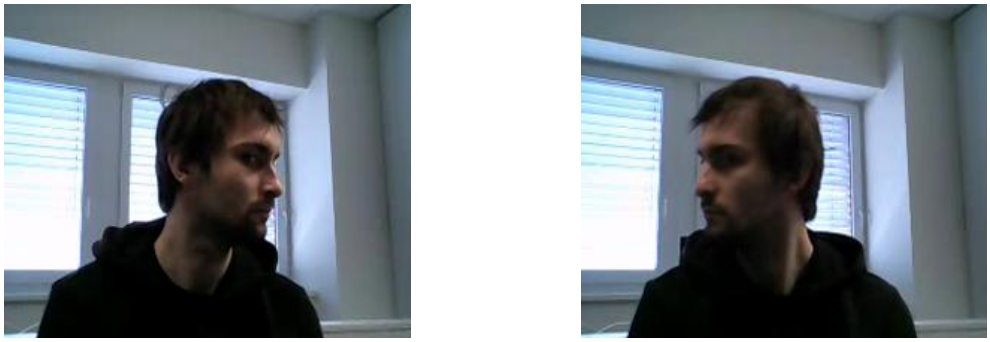
Dále se testovalo, jestli se obličej detekuje i v případě různého natočení. Pokud je obličej natočený přímo ke kameře je detekce úspěšná viz obrázek 6.8.



a) b)

Obrázek 6.8 Obličej natočený ke kameře

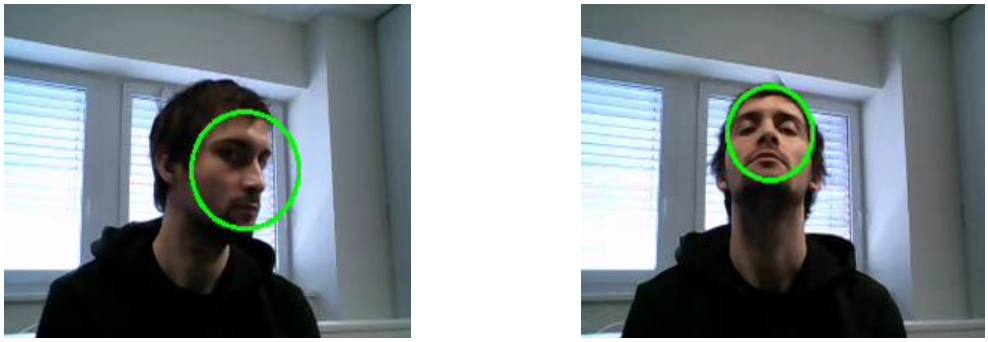
Pokud je obličej z profilu už k detekci nedochází. To znázorňují dílčí obrázky 6.9a a 6.9b, kde není vyznačená dekovaná oblast.



a) b)

Obrázek 6.9 Obličej z profilu

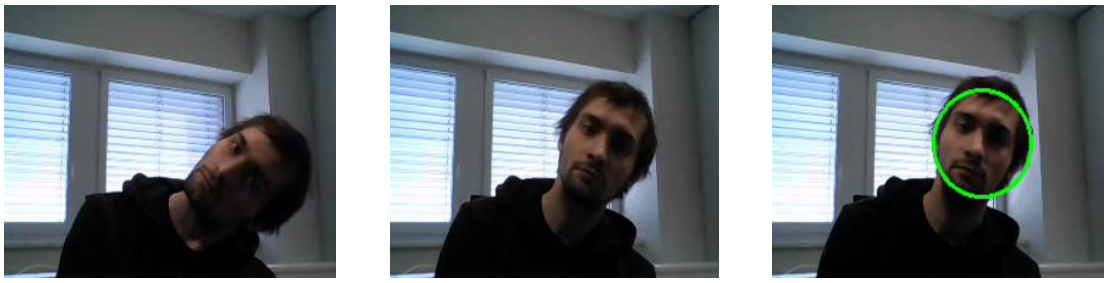
Na dalším obrázku 6.10 jsou zobrazeny hraniční polohy obličeje pro jeho detekci.



a) b)

Obrázek 6.10 Hraniční natočení obličeje pro detekci

Dále bylo vyzkoušeno, jestli k detekci dochází při naklonění. Pokud je obličej nakloněný na stranu k detekci nedochází viz obrázek 6.11. Na posledním dílčím snímku 6.11c je zobrazeno hraniční naklonění, kdy k detekci ještě došlo.



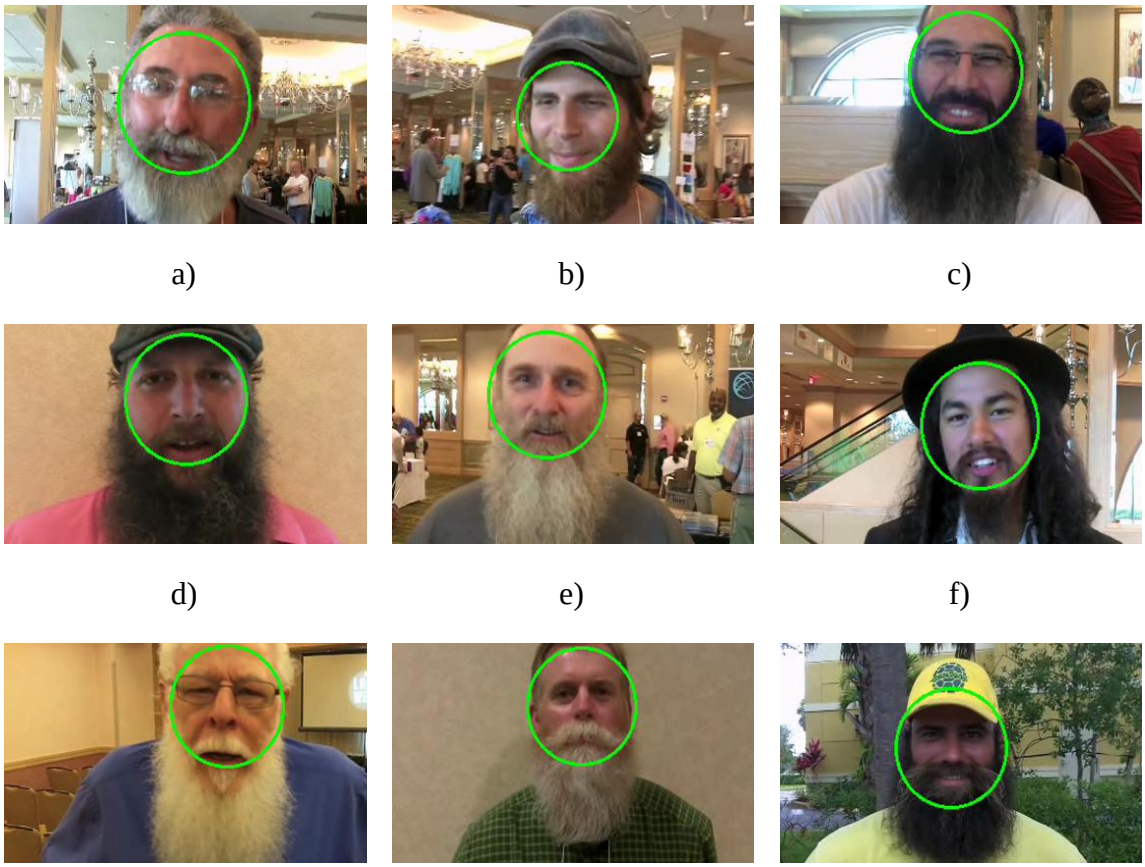
a) b) c)

Obrázek 6.11 Obličej natočený na stranu

Z těchto testů plyne, že je detekce obličeje úspěšná v případě nízkého natočení ve všech třech osách od polohy obličeje natočeného přímo ke kameře.

6.2.3 Obličej s vousy

Dalším testem se zkoušela detekce obličeje s vousy. I v tomto případě docházelo k detekci viz obrázek 6.12.



a) b) c)

d) e) f)

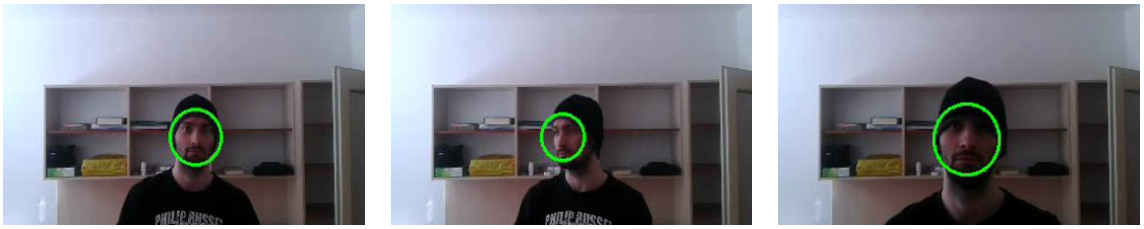
g) h) i)

Obrázek 6.12 Obličej s vousy [40]

Testované snímky byly pořízeny z videa. Zde byl obličej u každé osoby z většiny času natočen ke kameře. V krátkých časových okamžicích došlo k probliknutí vyznačené oblasti a tudíž ve snímku nebyl detekován obličej. Stalo se tak pouze při natočení obličeje na stranu nebo při přiblížení kamery a obličej byl pak z části oříznut. Tyto neúspěšné detekci byly minimální a téměř z velké části záznamu bylo pozorována vyznačená oblast.

6.2.4 Pokrývka hlavy

Dalším hlediskem úspěšnosti detekce může být situace, že má osoba pokrývku hlavy, jako je čepice nebo klobouk. Pokud snímek splňuje podmínky rozlišení a úhlu natočení obličeje je i v tomto případě detekce úspěšná viz obrázek 6.13 a 6.14.



a) b) c)

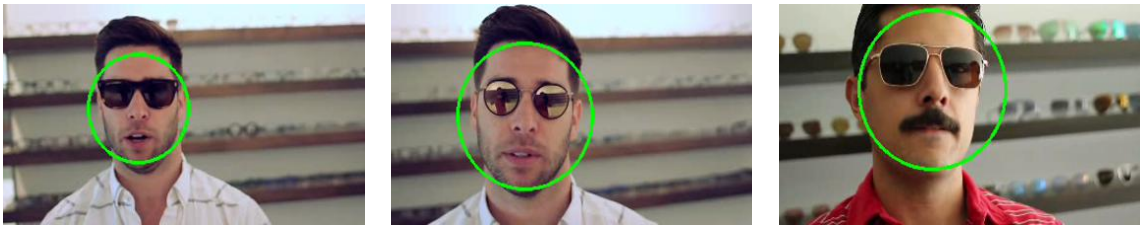
Obrázek 6.13 Obličej s pokrývkou hlavy č.1

a) b) c)

Obrázek 6.14 Obličej s pokrývkou hlavy č.2 [41]

6.2.5 Sluneční brýle

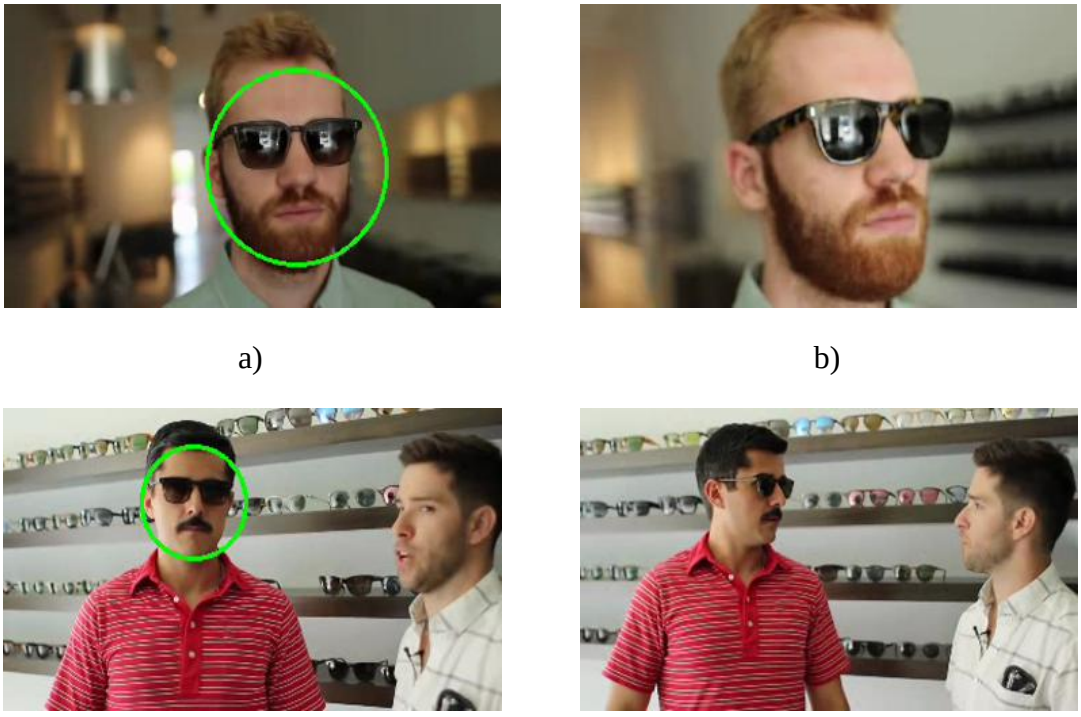
Dále byl obličej testován se slunečními brýlemi. Jak jde vidět na obrázku 6.15 i v případě slunečních brýlí došlo k úspěšné detekci.



a) b) c)

Obrázek 6.15 Obličej se slunečními brýlemi [42]

Pokud ale nebyla splněna podmínka natočení obličeje tak k detekci nedocházelo. Na dílčích obrázcích 6.16b a 6.16d je zobrazena neúspěšná detekce, protože je zde obličej z profilu. Obličej musí být více natočený ke kameře, aby mohl být detekován jak znázorňují obrázky 6.16a a 6.16c.



a) b)

c) d)

Obrázek 6.16 Natočený obličej se slunečními brýlemi [42]

6.2.6 Falešné detekce

Při detekci obličeje pomocí Haarových znaků může docházet i k falešným detekcím. To je zřejmě způsobeno tím, že je v obraze nalezena nějaká podobnost s obličejem. Předchozí testy detekce obličeje byly testovány převážně na videích, kde občas k těmto falešným detekcím docházelo.

Celkem se testovalo na pěti videích s celkovým počtem 29575 snímků kde došlo ke 192 falešných detekcí při výšce řádků snímku 200p. Procentuálně to je 0,6% snímků s falešnou detekcí. Některé falešné detekce jsou zobrazeny na obrázku 6.17.

a) b) c)

d) e) f)

Obrázek 6.17 Falešné detekce obličeje

6.3 Detekce pomocí Haarových znaků - horní část těla

Dále se testovala detekce horní části těla za použití souboru *haarcascade_upperbody.xml*. Tyto testy jsou méně podrobnější než u detekce obličeje. Jsou zaměřeny pouze na otestování jejich funkčnosti. Doba zpracování detekce horní části těla trvá přibližně 180ms při počtu řádků ve snímku 200p. Otestování lze vidět na obrázku 6.18.

a) b) c)

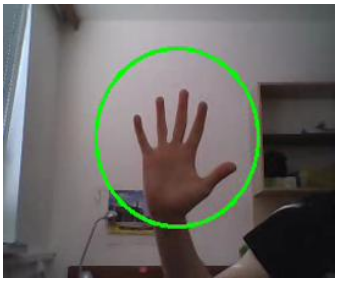
d) e) f)

Obrázek 6.18 Detekce horní části těla

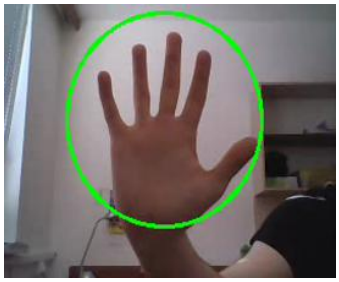
Z dílčích snímků je patrné, že k detekci nedochází vždy jak zobrazují obrázky 6.18a a 6.18f. Když je postava na okraji snímku a nebo moc blízko, tak k detekci nedochází.

6.4 Detekce pomocí Haarových znaků - ruka

Detekce ruky byla testována stejným způsobem jako detekce v předchozí podkapitole. Testovaným souborem byl *haarcascade_hand.xml*. Doba zpracování se pohybuje okolo 60 ms při počtu řádků ve snímku 200p. Na obrázku 6.19 je zobrazeno testování.

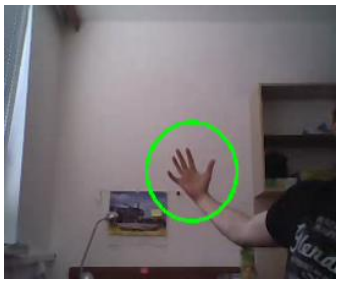


a)



b)

c)



d)

Obrázek 6.19 Detekce ruky

Testy byly zaměřeny v závislosti na vzdálenosti od kamery. Na obrázku 6.19c nebyla ruka detekována, protože byla moc blízko ke kameře. Naopak byla detekována i ve větší vzdálenosti viz obrázek 6.19d. A na obrázku 6.19b je zobrazeno maximální přiblížení pro detekování.

7 ZÁVĚR

Byly prostudovány metody zpracování obrazu a vyzkoušeny algoritmy využívajících funkcí knihoven OpenCV. Postupem jednotlivých detekcí bylo načíst vstupní data v podobě snímků, dále je zpracovat a vyhodnotit.

Realizoval se způsob k přístupu záznamů na zařízení a jeho ovládání v podobě webové stránky. Bylo vyzkoušeno, že v případě docházených detekcí se snímky zaznamenávají a informace o záznamech se ukládají do databáze. Pomocí záznamů v databázi se snímky zobrazují na webové stránce a je možné si je procházet. V případě nedostatku úložného místa se záznamy dají odstranit k uvolnění pro další záznamy.

Dále se provedlo testování systému. Detekce pohybu je úspěšná v případě správného nastavení parametrů a za předpokladu statické kamery. U detekce pomocí Haarových znaků byly otestovány tři soubory pro zpracování a to pro obličej, horní část těla a ruku. Detekce se tímto způsobem také podařily úspěšně prověřit. Fungují za předpokladu splnění určitých podmínek jako je pro obličej jeho natočení z pohledu kamery.

LITERATURA

[1] *Procesory ARM: Základ nové éry* [online]. [cit. 2016-12-14]. Dostupné z: http://www.zive.cz/clanky/procesory-arm-zaklad-nove-ery/sc-3-a-164061/default.aspx

[2] ARM. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2016-12-14]. Dostupné z: https://cs.wikipedia.org/wiki/ARM

[3] *Raspberry_PI3* [online]. In: . [cit. 2017-05-29]. Dostupné z: https://www.rpibolt.hu/shop_ordered/15922/pic/kepek/Raspberry_PI3/PI3-1.png

[4] *Raspberry Pi mění svět* [online]. [cit. 2016-12-14]. Dostupné z: http://tech.ihned.cz/geekosfera/c1-65195330-raspberry-pi-meni-svet-seznamte-se-s-nejzajimavejsim-pocitacem-dneska

[5] Raspberry Pi. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2016-12-14]. Dostupné z: https://cs.wikipedia.org/wiki/Raspberry_Pi

[6] *O barevných modelech RGB a CMYK* [online]. [cit. 2016-12-14]. Dostupné z: http://www.digineff.cz/clanek/pojmy/o-barevnych-modelech-rgb-cmyk

[7] *Detekce obličeje v obraze* [online]. [cit. 2016-12-14]. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=85175

[8] *Segmentace obrazu* [online]. [cit. 2016-12-14]. Dostupné z: http://is.muni.cz/th/72784/fi_m/dp.pdf

[9] Segmentace obrazu: Prahování. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2016-12-14]. Dostupné z: https://cs.wikipedia.org/wiki/Segmentace_obrazu

[10] *Morfologické operace* [online]. [cit. 2016-12-14]. Dostupné z: http://midas.uamt.feec.vutbr.cz/ZVS/Exercise10/content_cz.php

[11] Morphological Image Processing. *Morphological Image Processing* [online]. 2000 [cit. 2017-05-29]. Dostupné z: https://www.cs.auckland.ac.nz/courses/compsci773s1c/lectures/ImageProcessing-html/topic4.htm

[12] *APLIKACE METOD DETEKCE A ROZPOZNÁNÍ OBLIČEJE* [online]. [cit. 2016-12-14]. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=82657

[13] *Detekce objektů v obraze s pomocí Haarových příznaků* [online]. Brno, 2012 [cit. 2017-05-30]. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=51400

[14] *Haar* [online]. In: . [cit. 2017-05-29]. Dostupné z: https://kosbr.github.io/images/articles/opencv/haar.jpg

[15] *Fig-2-Haar* [online]. In: . [cit. 2017-05-29]. Dostupné z: https://www.researchgate.net/profile/Tomas_Proscevicius2/publication/237049645/figure/fig1/AS:299454470082567@1448406917463/Fig-2-Haar-features-examples-for-face-detection.png

[16] *Obecné informace o knihovně* [online]. [cit. 2016-12-14]. Dostupné z: https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=25212

[17] *Rozšíření knihovny pro zpracování obrazu* [online]. [cit. 2016-12-14]. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=27890

[18] *OpenCV* [online]. In: . [cit. 2017-05-29]. Dostupné z: https://www.researchgate.net/profile/Mukul_Shirvaikar/publication/252973402/figure/fig2/AS:298216210878465@1448111693053/Figure-2-The-basic-structure-of-OpenCV.png

[19] *Learning OpenCV* [online]. [cit. 2016-12-14]. Dostupné z: http://www.bogotobogo.com/cplusplus/files/OReilly%20Learning%20OpenCV.pdf

[20] *OpenCV | OpenCV: OpenCV API Reference* [online]. [cit. 2016-12-14]. Dostupné z: http://docs.opencv.org/2.4/modules/refman.html

[21] *Detekce hran, segmentace* [online]. [cit. 2016-12-14]. Dostupné z: https://kalabovi.org/pitel:msz:detekce_hran_segmentace

[22] *Základy spracovania obrazu* [online]. In: . [cit. 2017-05-25]. Dostupné z: http://www.sccg.sk/~madaras/iip/hough_1.jpg

[23] Cannyho hranový detektor. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-05-29]. Dostupné z: https://cs.wikipedia.org/wiki/Cannyho_hranov%C3%BD_detektor

[24] *Detekce pohybu ve videu a jejich identifikace* [online]. [cit. 2016-12-14]. Dostupné z: http://fbmi.cvut.cz/files/predmety/3528/public/Detekce%20pohybu%20ve%20videu.pdf

[25] Dublin Life. In: *Youtube* [online]. [cit. 2017-05-29]. Dostupné z: https://www.youtube.com/watch?v=q-1YkH8RHdQ

[26] *Analýza obrazu* [online]. [cit. 2016-12-14]. Dostupné z: http://midas.uamt.feec.vutbr.cz/POV/LPOV_Exercise01/content_cz.php

[27] *Lena* [online]. In: . [cit. 2017-05-29]. Dostupné z: https://www.cosy.sbg.ac.at/~pmeerw/Watermarking/lena_color.gif

[28] HyperText Markup Language. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-05-29]. Dostupné z: https://cs.wikipedia.org/wiki/HyperText_Markup_Language

[29] Jak psát web: Bloky. *Jak psát web: Bloky* [online]. [cit. 2017-05-29]. Dostupné z: https://www.jakpsatweb.cz/html/bloky.html#div

[30] Kaskádové styly. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-05-29]. Dostupné z: https://cs.wikipedia.org/wiki/Kask%C3%A1dov%C3%A9_styly

[31] PHP. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-05-29]. Dostupné z: https://cs.wikipedia.org/wiki/PHP

[32] JavaScript. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-05-29]. Dostupné z: https://cs.wikipedia.org/wiki/JavaScript

[33] PhpMyAdmin. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-05-29]. Dostupné z: https://cs.wikipedia.org/wiki/PhpMyAdmin

[34] Nest Cam. In: *Youtube* [online]. [cit. 2017-05-29]. Dostupné z: https://www.youtube.com/watch?v=kjRmgDJC3ZM

[35] [online]. In: . [cit. 2017-05-29]. Dostupné z: https://s-media-cache-ak0.pinimg.com/736x/cc/ec/ba/ccecba1e53cc1f9f5608c95689c028e2.jpg

[36] *Oscars* [online]. In: . [cit. 2017-05-29]. Dostupné z:

https://ewedit.files.wordpress.com/2015/02/oscars-ellen-selfie_0.jpg?w=612

[37] *Group of happy people* [online]. In: . [cit. 2017-05-29]. Dostupné z: https://previews.123rf.com/images/kurhan/kurhan1206/kurhan120600190/13977894-Group-of-happy-people-Stock-Photo-family-happy-dental.jpg

[38] *Pexels-photo* [online]. In: . [cit. 2017-05-29]. Dostupné z: https://static.pexels.com/photos/109919/pexels-photo-109919.jpeg

[39] *People* [online]. In: . [cit. 2017-05-29]. Dostupné z: https://media.licdn.com/mpr/mpr/AAEAAQAAAAAAAAyNAAAAJDRkMjJmNmY1LWM0MDMtNGYzOS1hMzU4LTA1YmU5NzA0ZDgzOQ.jpg

[40] Beard. In: *Youtube* [online]. [cit. 2017-05-29]. Dostupné z: https://www.youtube.com/watch?v=rySd2kYvzMM

[41] Hairstyle. In: *Youtube* [online]. [cit. 2017-05-29]. Dostupné z: https://www.youtube.com/watch?v=ZYW2piyJafU

[42] Sunglasses. In: *Youtube* [online]. [cit. 2017-05-29]. Dostupné z: https://www.youtube.com/watch?v=WcIFMaqEUX0